

# Specification for the CMS Hadron Calorimetry Front End Readout Module Channel Control ASIC

Theresa Shaw, Alan Baumbaugh, Ahmed Boubekeur, John Elias

November 23, 1999

Revised

March 8, 2002

## Revision History:

09/14/2000	TMS	Fixed Fig.1 and Fig. 2 to show clocking coming from Master Clock source and not CCA. Fixed Data Format and Orbit Message format to match up with CHFET Serializer specification. Added "Send Fill Frame" bit to Control Register.
09/26/2000	TMS	Simplify CAP ID checking – no more voting logic. Just flag CAP ID mismatch and flag error. Added detail. Added pinout table. Added simulaion ideas.
10/30/2000	TMS	Changed: Serial bus named RBXbus. Worst case channel to channel spread changed from 15ns to 58ns. Appendix B – this logic suggested by Al Baumbaugh for the alignment of the data from the two QIEs serviced by the CCA. Added some pin info – we now have 85 signal pins Added Beam Zero timing info Added second QIE_RESET pulse, so that each QIE channel has its own - implementaion of QIE_Rest is described in Appendix B Added an RBXbus register to control alignment mux's described in Appendix B Test Pulses changed from 10 clock cycle width to 1 clock cycle width AND polarity control. Also, these pulses must be timed in the same way as the QIE_RESET pulses, see APPENDIX B. Orbit Message is now sent upon the receipt of Beam_zero. Previously, it worked off a counter.
1/9/2001	TMS	Redo Serializer Output section to match CERN GOL ASIC Both CAP Ids will be drive out of CCA RBXbus Address pins RBX_A(7:2) all programmable Polarity change of Send_Fill_Frame signal Add fill frames to Orbit message
7/12/2001	TMS	Add pinout information
2/12/2002	TMS	Revise pinout information; new package 120 pin 14mmx20mm; rename some of "die" pad names to make PCB routing easier. For instance, Mant 0 thru 4, renamed Mant 4 thru 0. This changes definition of TXD output bits.
2/19/2002	TMS	revise pinout to new package 128pin 14mm x 20mm
3/08/2002	TMS	revise pinout to add package pin for leadframe die pad

## Introduction

The CMS Hadron Calorimetry Front End Readout Module is based upon the specification, design and production of several ASIC devices.

The FE Module will receive inputs from six Hybrid Photodiodes (HPDs). Each of these inputs will be fed into a “QIE” charge integrating ASIC which has a built in Flash ADC. The digital data resulting from the conversion is then passed to the Channel Control ASIC.

Figure 1 illustrates the connections between the Channel Control ASIC and other FE module components. Figure 2 is a block diagram of the Channel Control ASIC.

Each Channel Control ASIC (CCA) receives the digital data from two QIEs. Each QIE presents a 5 bit mantissa, a 2 bit exponent and a 2 bit Capacitor ID.

The CCA must provide the following functions:

- The processing and synchronization of the data from two QIEs,
- The provision of QIE clocking signals to run the QIE charge integrator and Flash ADC,
- Checking of the accuracy of the Capacitor IDs (the Cap IDs from different QIEs should be in synchronization),
- The ability to force the QIE to use a given range,
- The ability to set Pedestal DAC values,
- The ability to issue two test pulse triggers of programmable polarity,
- The provision of event synchronization checks – a crossing counter will be implemented and checked for accuracy with every beam turn marker,
- The ability to send a known pattern to the serial optic link,
- The ability to “reset” the QIE,
- And, the ability to send and report on any detected errors at a known and determined time.

A serial link will be implemented on the Channel Control ASIC to allow users to download information which controls its operation.

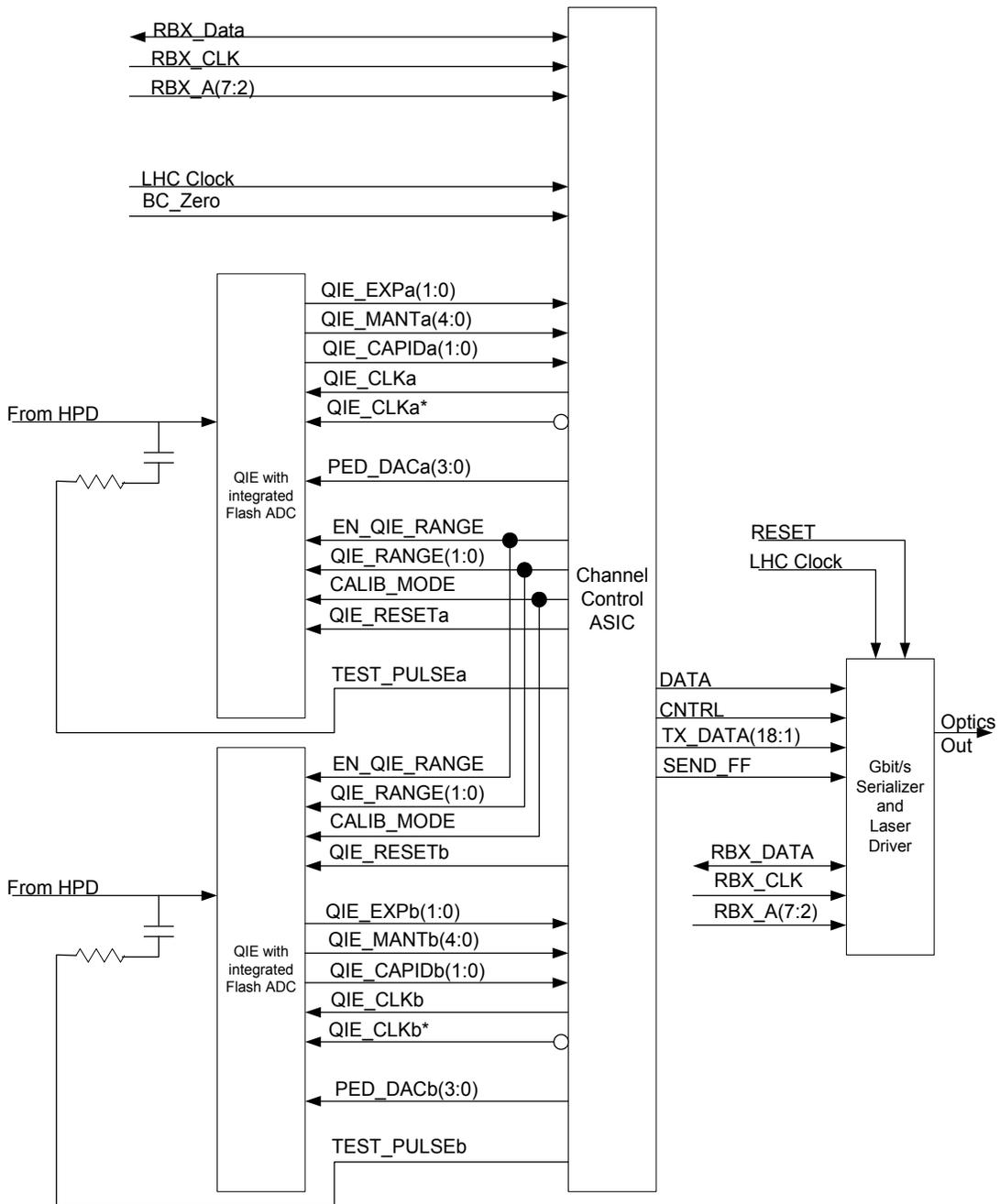


Figure 1 Channel Control ASIC Connections

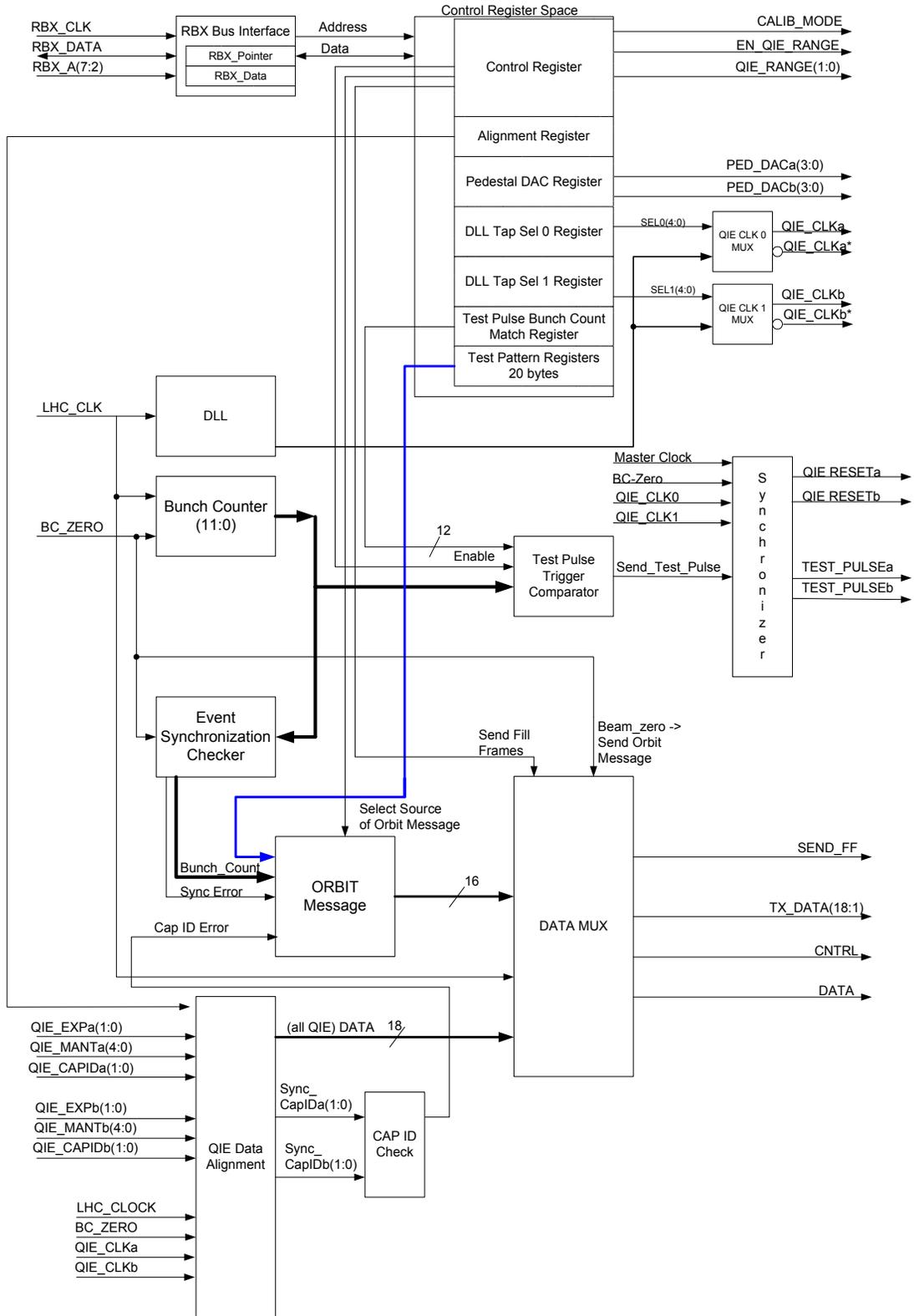


Figure 2 Channel Control ASIC Block Diagram

## The RBXbus and Register Access

The RBX bus protocol will be used to communicate with the Channel Control ASIC. The RBX bus is an asynchronous 2-wire serial bus and allows for read/write operations at up to 3.4 Mbits/s.

### Data Transfer via the RBXbus

The RBXbus provides a seven bit address field. To provide communication with the CCA, each CCA will implement two RBXbus registers. The first register will be an address pointer, **RBX\_Pointer**. This will be an seven bit register which is loaded with an address pointer to register space.

The second RBXbus register implemented by each CCA will be an eight bit **RBX\_Data** register. The data written to this register will actually be written to the register space which is pointed to by the **RBX\_Pointer**. Likewise, data read from the **RBX\_Data** register will come from the register being pointed to by the **RBX\_Pointer**.

Thus; to accomplish communication with the CCA, only two RBXbus registers must be addressable on each device. Six "ID" or programmable address pins, **RBX\_A(7:2)**, will be used to set the individual CCA addresses. **RBX\_A1** will be used to determine whether the **RBX\_Pointer** or the **RBX\_Data** register is being addressed.

The RBXbus addresses for the two registers will be set as follows:

### RBXbus addresses

RBXbus Register	A7	A6	A5	A4	A3	A2	A1
RBX_Pointer	<b>RBX_A7</b>	<b>RBX_A6</b>	<b>RBX_A5</b>	<b>RBX_A4</b>	<b>RBX_A3</b>	<b>RBX_A2</b>	<b>0</b>
RBX_Data	<b>RBX_A7</b>	<b>RBX_A6</b>	<b>RBX_A5</b>	<b>RBX_A4</b>	<b>RBX_A3</b>	<b>RBX_A2</b>	<b>1</b>

## Register Space

The following registers are read and written through the RBX serial bus interface. Default power on value should be “low” in all cases.

### Control Register

General Description: The Control Register contains bits to turn on/off various CCA functions.

RBXbus Pointer Address = 0x00

<b><u>Bit</u></b>	<b><u>Definition</u></b>
7	Polarity of Test Pulse 0 – Test Pulse will be high going pulse 1 – Test Pulse will be low going pulse
6	Send Fill Frames <i>This bit, when set, drives the output signal “Send_Fill_Frames” high.</i>
5	Enable Calibration Mode
4	QIE Force Range(1)
3	QIE Force Range(0)
2	Enable QIE Force Range <i>This bit will cause the QIE to stop autoranging, and use the range indicated though the QIE Force Range(1:0) bits.</i>
1	Select Orbit Data <i>This bit controls which of the two sources is selected for the Orbit Message.</i> 0 – Orbit message will consist of one word of Cap ID synchronization checking, one word of Bunch Counter synchronization checking and 8 words of fill frames. 1 – Orbit message will be the 20 bytes of Test Pattern Data packed as 16-bit words.
0	Enable Test Pulse <i>This bit will enable the sending of a Test Pulse when matches occur with the Test Pulse Bunch Count Match Register and the Beam Counter.</i>

## Alignment Control Channel a Register

General Description: The Alignment Control Register contains bits control the channel 0 alignment feature of the CCA.

**Refer to Appendix B for explanation of these settings.**

RBXbus Pointer Address = 0x01

<b><u>Bit</u></b>	<b><u>Definition</u></b>
7	Undefined
6:5	Selects phase of Channel 0 Data <b><i>(MUX D in Appendix B)</i></b> 0 – selects Q0D2 1 – selects Q0D3 2 – selects Q0D4 3 – undefined
4	Master_Clock polarity select – for data alignment of Channel 0 <b><i>(Mux C in Appendix B)</i></b> 0 – selects inverse of Master_Clock 1 – selects Master_Clock
3	Undefined
2:1	Selects phase of QIE_RESET0 or TEST_PULSE_TRIG0 signal <b><i>(MUX B setting in Appendix B)</i></b> 0 – selects Q0R2 1 – selects Q0R3 2 – selects Q0R4 3 – undefined
0	QIE_CLK0 polarity select for aligning QIE control signal QIE_RESET0 or TEST_PULSE_TRIG0 <b><i>(MUX A setting in Appendix B)</i></b> 0 – selects inverse of QIE_CLK0 1 – selects QIE_CLK0

## Alignment Control Channel b Register

General Description: The Alignment Control Register contains bits control the channel 1 alignment feature of the CCA.

**Refer to Appendix B for explanation of these settings.**

RBXbus Pointer Address = 0x02

<b><u>Bit</u></b>	<b><u>Definition</u></b>
7	Undefined
6:5	Selects phase of Channel 1 Data <b><i>(MUX D in Appendix B)</i></b> 0 – selects Q0D2 1 – selects Q0D3 2 – selects Q0D4 3 – undefined
5	Master_Clock polarity select – for data alignment of channel 1 <b><i>(Mux C in Appendix B)</i></b> 0 – selects inverse of Master_Clock 1 – selects Master_Clock
3	Undefined
2:1	Selects phase of QIE_RESET1 or TEST_PULSE_TRIG1 signal <b><i>(MUX B setting in Appendix B)</i></b> 0 – selects Q0R2 1 – selects Q0R3 2 – selects Q0R4 3 – undefined
0	QIE_CLK1 polarity select for aligning QIE control signal QIE_RESET1 or TEST_PULSE_TRIG1 <b><i>(MUX A setting in Appendix B)</i></b> 0 – selects inverse of QIE_CLK1 1 – selects QIE_CLK1

### **Pedestal DAC Register**

General Description: The Pedestal DAC Register contains the settings to two 4-bit DAC settings.

RBXbus Pointer Address = 0x03

<b><u>Bit</u></b>	<b><u>Definition</u></b>
7	DACb Pedestal Value (3)
6	DACb Pedestal Value (2)
5	DACb Pedestal Value (1)
4	DACb Pedestal Value (0)
3	DACa Pedestal Value (3)
2	DACa Pedestal Value (2)
1	DACa Pedestal Value (1)
0	DACa Pedestal Value (0)

### **DLL Tap Select 0 Register**

General Description: The DLL Tap Select 0 Register contains the select code for the DLL phase chosen to run the QIE clock for channel 0. The DLL must allow for a 25ns shift of the clock phase in increments of 1ns.

RBXbus Pointer Address = 0x04

<b><u>Bit</u></b>	<b><u>Definition</u></b>
7	
6	
5	
4	Select (4)
3	Select (3)
2	Select (2)
1	Select (1)
0	Select (0)

### DLL Tap Select 1 Register

General Description: The DLL Tap Select 1 Register contains the select code for the DLL phase chosen to run the QIE clock for channel 1.

RBXbus Pointer Address = 0x05

<u>Bit</u>	<u>Definition</u>
7	
6	
5	
4	Select (4)
3	Select (3)
2	Select (2)
1	Select (1)
0	Select (0)

### Test Pulse Bunch Count Match Registers

General Description: The Test Pulse Bunch Count Match Register contains the Bunch Count at which a Test Pulse should be fired, providing the “Enable Test Pulse” bit has been set in the Control Register.

RBXbus Pointer Address = 0x06 (MSB)

RBXbus Address = 0x07 (LSB)

<u>Bit</u>	<u>Definition</u>	<u>Bit</u>	<u>Definition</u>
7		7	Data (7)
6		6	Data (6)
5		5	Data (5)
4		4	Data (4)
3	Data (11)	3	Data (3)
2	Data (10)	2	Data (2)
1	Data (9)	1	Data (1)
0	Data (8)	0	Data (0)

### Test Pattern Registers

General Description: This Register Block contains 10 16-bit words which can be the source of the Orbit Message whenever the “Select Orbit Data” bit in the Control Register is set “high”.

RBXBUS Pointer Address = 0x08 thru 0x21

## Register Summary

<b>RBXBUS Pointer</b>	<b>Register Name</b>
0x00	Control
0x01	Alignment Control Channel a
0x02	Alignment Control Channel b
0x03	Pedestal DAC
0x04	DLL Tap Select 0
0x05	DLL Tap Select 1
0x06	Test Pulse Bunch Count Match (LSB)
0x07	Test Pulse Bunch Count Match (MSB)
0x08	Test Pattern Byte 0
0x09	Test Pattern Byte 1
0x0A	Test Pattern Byte 2
0x0B	Test Pattern Byte 3
0x0C	Test Pattern Byte 4
0x0D	Test Pattern Byte 5
0x0E	Test Pattern Byte 6
0x0F	Test Pattern Byte 7
0x10	Test Pattern Byte 8
0x11	Test Pattern Byte 9
0x12	Test Pattern Byte 10
0x13	Test Pattern Byte 11
0x14	Test Pattern Byte 12
0x15	Test Pattern Byte 13
0x16	Test Pattern Byte 14
0x17	Test Pattern Byte 15
0x18	Test Pattern Byte 16
0x19	Test Pattern Byte 17
0x1A	Test Pattern Byte 18
0x1B	Test Pattern Byte 19

## **DLL**

This block produces a set of 32 1ns taps which are phased to the Master Clock. The 32 ns delay is derived from 25 1-ns taps of the DLL, plus the ability to invert the clock phase, which adds another possible 12 1-ns steps.

### **QIE CLKa Mux**

This block will select a properly phased QIE CLK from one of the 32 possible 1ns steps available from the DLL. The selected clock signal should be driven as differential LVDS to QIE channel 0.

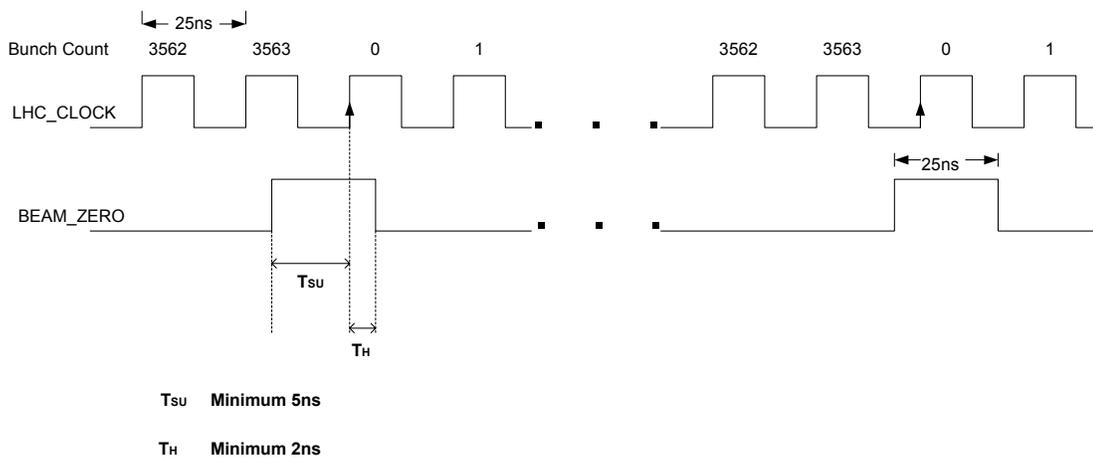
### **QIE CLKb Mux**

This block will select a properly phased QIE CLK from one of the 32 possible 1ns steps available from the DLL. The selected clock signal should be driven as differential LVDS to QIE channel 1.

## Bunch Counter

This block counts the number of bunches. It increments with LHC\_CLOCK and will be reset with each BC\_ZERO. The final count prior to BC\_ZERO must be latched and included in the Orbit Message. It must also be checked against the expected number to verify the correct functioning of the counter for each turn. If an error is detected, a flag bit must be set and included in the Orbit Message.

The counter must be 12 bits wide.



The Bunch Count should be reset on rising edge of LHC\_CLOCK whenever Beam Zero signal is present

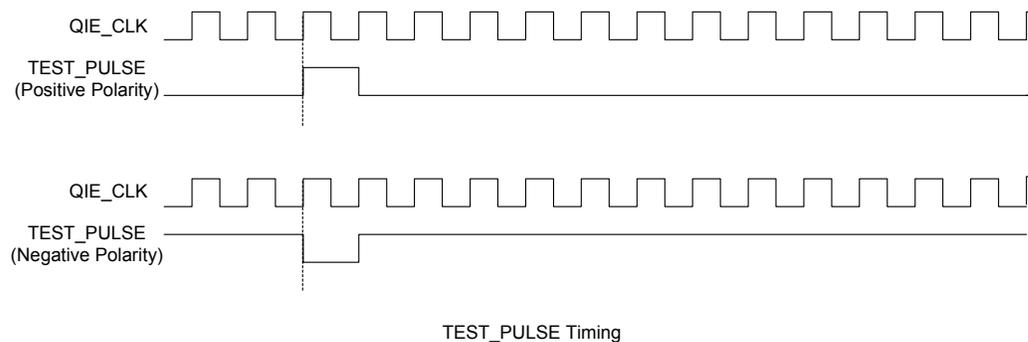
## Bunch Counter for CCA

## Test Pulse Trigger Comparator

This logic produces two Test Pulse Triggers which are triggered when the “Enable Test Pulse” bit is set in the Control Register and the Bunch Counter matches the Test Pulse Bunch Count Match Registers.

The Test Pulse produced should be 1 clock cycle long and in sync with the QIE clock pulse derived from the DLL and clock mux. The polarity of the Test Pulse is set through a bit in the Control Register.

Additionally, the Test Pulse must go through an alignment cycle similar to the QIE\_RESET (see Appendix B). It will use the same mux settings as the QIE\_RESET Pulse.



## Event Synchronization Checker

This logic must verify that the final value of the Bunch counter, prior to reset by the BC\_ZERO signal, matched the expected value. If it does not, the error condition must be flagged in the Orbit Message.

The Bunch counter should count from 0 to 3563.

## CAP ID Check

The width of the data field sent to the serializer chip allows for 16/32 bits of information. This will include 5 bits of mantissa and 2 bits of exponent from each of the two QIE channels, accounting for 14 bits. Since we may not have the space to send on the CAP IDs from each channel, we must select a CAP ID to represent both channels. Because the CAP IDs should match in all cases unless an error or a SEU has occurred, this is not unreasonable.

The CAP ID Check block will compare the CAP IDs which come from two QIE channels. If the CAP IDs from the two QIEs do not match, the CAP ID Check logic will flag a CAP ID error. The CAP ID error bit will be included in the Orbit Message. In all

cases, the 2 bits of CAP ID sent to the serializer are from channel 0. However, the CAP IDs from both channels will be driven out of the CCA as part of the 18 bit output data field.

### QIE Data Alignment

This block will guarantee that the two channels of QIE are properly aligned.

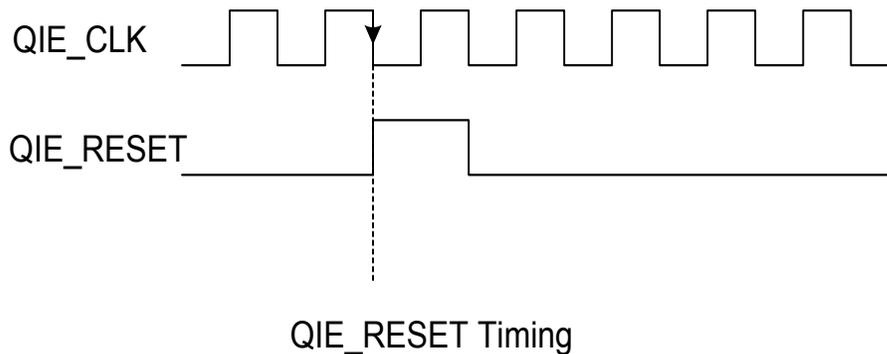
The worst case misalignment will be 58 ns.

QIE data is clock into the CCA on the rising edge of the LHC\_Clock.

**Appendix B presents a suggested logical implementation for the alignment block.**

### Synchronization

This block will guarantee that the QIE control signals, QIE\_RESET and TEST\_PULSE\_TRIG, are properly timed.



**Appendix B presents a suggested logical implementation for the alignment block.**

### Data Multiplexer

This block sends the data out to the gigabit serializing chip. It must choose to send either detector data, the Orbit Message, or a series of Fill Frames (to synchronise the link).

Data is clocked out of the CCA on the rising edge of the LHC Clock.

## **Orbit Message**

The Orbit message should be sent whenever a Beam Zero is detected.

This block must control the assembling and sending of the Orbit Message. It receives inputs from the CAP ID checker, the Event Synchronization Checker and the Test Pattern Registers.

The source of data for the Orbit Message is controlled by the “Select Orbit Data” bit of the Control Register.

If the “Select Orbit Data” bit is low, the source of the Orbit Message will be the error information from the CAP ID checker and Bunch Counter synchronization checker, followed by fill frames.

If the “Select Orbit Data” bit is high, the source of the Orbit Message will be the Test Pattern Registers, followed by fill frames.

## Data Format

The following format should be used when sending data to the gigabit serializing IC.

<b>Data Out</b>	<b>Signal Name</b>
<b>Send Fill Frame</b>	<b>0</b>
<b>DATA</b>	<b>1</b>
<b>CNTRL</b>	<b>0</b>
<b>Data(18)</b>	<b>CAP_ID_Chan1(1)</b>
<b>Data(17)</b>	<b>CAP_ID_Chan1(0)</b>
<b>Data(16)</b>	<b>QIE_Chan1_Mant(0)</b>
<b>Data(15)</b>	<b>QIE_Chan1_Mant(1)</b>
<b>Data(14)</b>	<b>QIE_Chan1_Mant(2)</b>
<b>Data(13)</b>	<b>QIE_Chan1_Mant(3)</b>
<b>Data(12)</b>	<b>QIE_Chan1_Mant(4)</b>
<b>Data(11)</b>	<b>QIE_Exp1(0)</b>
<b>Data(10)</b>	<b>QIE_Exp1(1)</b>
<b>Data(9)</b>	<b>CAP_ID_Chan0(1)</b>
<b>Data(8)</b>	<b>CAP_ID_Chan0(0)</b>
<b>Data(7)</b>	<b>QIE_Chan0_Mant(0)</b>
<b>Data(6)</b>	<b>QIE_Chan0_Mant(1)</b>
<b>Data(5)</b>	<b>QIE_Chan0_Mant(2)</b>
<b>Data(4)</b>	<b>QIE_Chan0_Mant(3)</b>
<b>Data(3)</b>	<b>QIE_Chan0_Mant(4)</b>
<b>Data(2)</b>	<b>QIE_Exp0(0)</b>
<b>Data(1)</b>	<b>QIE_Exp0(1)</b>

## Orbit Message

If, “Select Orbit Data” bit of the Control Register = 0,

<b>Data Out</b>	<b>Word 0 Event_Sync</b>	<b>Words 1-69 Fill Frames (See Note 1)</b>
<b>Send_Fill_Frame</b>	0	1
<b>DATA</b>	0	X
<b>CNTRL</b>	1	X
<b>Data(18)</b>	X	X
<b>Data(17)</b>	X	X
<b>Data(16)</b>	X	X
<b>Data(15)</b>	X	X
<b>Data(14)</b>	CAP ID Error	X
<b>Data(13)</b>	BNCH_CNT Error	X
<b>Data(12)</b>	Bnch_Cnt(11)	X
<b>Data(11)</b>	Bnch_Cnt(10)	X
<b>Data(10)</b>	Bnch_Cnt(9)	X
<b>Data(9)</b>	Bnch_Cnt(8)	X
<b>Data(8)</b>	Bnch_Cnt(7)	X
<b>Data(7)</b>	Bnch_Cnt(6)	X
<b>Data(6)</b>	Bnch_Cnt(5)	X
<b>Data(5)</b>	Bnch_Cnt(4)	X
<b>Data(4)</b>	Bnch_Cnt(3)	X
<b>Data(3)</b>	Bnch_Cnt(2)	X
<b>Data(2)</b>	Bnch_Cnt(1)	X
<b>Data(1)</b>	Bnch_Cnt(0)	X

**Note 1:** The output signal “Send\_Fill\_Frames” must be driven high when Fill Frames are sent.

CAP ID Error = 0; no error occurred  
 = 1; error detected

BNCH\_CNT Error =0; no error occurred  
 =1; error detected

Bnch\_Cnt(11:0) is the number in the Bunch Counter prior to receiving the Beam\_Zero Reset.

Or, if, “Select Orbit Data” bit of the Control Register = 1,  
the 20 bytes of Test Pattern Data should be sent.

<b>Word</b>	<b>SEND FILL FRAME</b>	<b>DATA</b>	<b>CNTRL</b>	<b>D18</b>	<b>D17</b>	<b>MSB(16:9)</b>	<b>LSB(8:1)</b>
0	0	1	0	X	X	Test Pattern 1	Test Pattern 0
1	0	1	0	X	X	Test Pattern 3	Test Pattern 2
2	0	1	0	X	X	Test Pattern 5	Test Pattern 4
3	0	1	0	X	X	Test Pattern 7	Test Pattern 6
4	0	1	0	X	X	Test Pattern 9	Test Pattern 8
5	0	1	0	X	X	Test Pattern B	Test Pattern A
6	0	1	0	X	X	Test Pattern D	Test Pattern C
7	0	1	0	X	X	Test Pattern F	Test Pattern E
8	0	1	0	X	X	Test Pattern 11	Test Pattern 10
9	0	1	0	X	X	Test Pattern 13	Test Pattern 12
10-69	1	X	X	X	X	X	X

**CCA Pinout Information**  
**120 PIN quad Flat Pack – 14mm X 20mm**  
**see Appendix C**

Pin #	Pin Name	Pad #	Pad Name	Description	Signal Type	Signal Level
1	N/C					
2	N/C					
3	N/C					
4	<b>QIE_MANTb(0)</b>	1	<b>QIE_MANTb(4)</b>	Bit(0) of QIE Mantissa – Channel_b (lsb)	Input	LVDS
5	<b>QIE_MANTb(0)*</b>	2	<b>QIE_MANTb(4)*</b>	Bit(0)* of QIE Mantissa – Channel_b (lsb)	Input	LVDS
6	QIE_CAPIDb(0)	3	QIE_CAPIDb(0)	Bit(0) of QIE Capacitor ID – Channel_b (lsb)	Input	LVDS
7	QIE_CAPIDb(0)*	4	QIE_CAPIDb(0)*	Bit(0)* of QIE Capacitor ID – Channel_b (lsb)	Input	LVDS
8	QIE_CAPIDb(1)	5	QIE_CAPIDb(1)	Bit(1) of QIE Capacitor ID – Channel_b (msb)	Input	LVDS
9	QIE_CAPIDb(1)*	6	QIE_CAPIDb(1)*	Bit(1)* of QIE Capacitor ID – Channel_b (msb)	Input	LVDS
10	VDD	7	VDD		pwr/gnd	3.3V
11	GND	8	GND		pwr/gnd	0V
12	QIE_CLKb	9	QIE_CLKb	Phase adjusted differential clock provided to QIE – Channel_b	Output	LVDS
13	QIE_CLKb*	10	QIE_CLKb*	Phase adjusted differential clock provided to QIE – Channel_b	Output	LVDS
14	QIE_CLKa	11	QIE_CLKa	Phase adjusted differential clock provided to QIE – Channel_a	Output	LVDS
15	QIE_CLKa*	12	QIE_CLKa*	Phase adjusted differential clock provided to QIE – Channel_a	Output	LVDS
16	LHC_CLK*	13	LHC_CLK*		Input	LVPE CL
17	LHC_CLK	14	LHC_CLK	40.08MHz clock	Input	LVPE CL
18	GND	15	GND		pwr/gnd	0V
19	QIE_RESEtb	16	QIE_RESEtb	Reset signal provided to the QIEs once an orbit – Channel_b	Output	3.3V CMOS
20	QIE_RESEta	17	QIE_RESEta	Reset signal provided to the QIEs once an orbit – Channel_a	Output	3.3V CMOS
21	TEST_PULSEb	18	TEST_PULSEb	Test Pulse produced by CCA which is sync'd	Output	3.3V CMOS

				with QIE_CLKb		
22	TEST_PULSEa	19	TEST_PULSEa	Test Pulse produced by CCA which is sync'd with QIE_CLKa	Output	3.3V CMOS
23	CALIB_MODE	20	CALIB_MODE	Forces QIE into Calibration Mode	Output	3.3V CMOS
24	QIE_RANGE(0)	21	QIE_RANGE(0)	Sets QIE Range (lsb) used when EN_QIE_RANGE is set high	Output	3.3V CMOS
25	QIE_RANGE(1)	22	QIE_RANGE(1)	Sets QIE Range (msb) used when EN_QIE_RANGE is set high	Output	3.3V CMOS
26	EN_QIE_RANGE	23	EN_QIE_RANGE	Forces QIE to use a given Range	Output	3.3V CMOS
27	PED_DACa(0)	24	PED_DACa(0)	Sets Pedestal DAC value driven to QIE – Channel a (lsb)	Output	3.3V CMOS
28	PED_DACa(1)	25	PED_DACa(1)	Sets Pedestal DAC value driven to QIE – Channel a	Output	3.3V CMOS
29	PED_DACa(2)	26	PED_DACa(2)	Sets Pedestal DAC value driven to QIE – Channel a	Output	3.3V CMOS
30	PED_DACa(3)	27	PED_DACa(3)	Sets Pedestal DAC value driven to QIE – Channel a (msb)	Output	3.3V CMOS
31	PED_DACb(0)	28	PED_DACb(0)	Sets Pedestal DAC value driven to QIE – Channel b (lsb)	Output	3.3V CMOS
32	PED_DACb(1)	29	PED_DACb(1)	Sets Pedestal DAC value driven to QIE – Channel b	Output	3.3V CMOS
33	PED_DACb(2)	30	PED_DACb(2)	Sets Pedestal DAC value driven to QIE – Channel b	Output	3.3V CMOS
34	PED_DACb(3)	31	PED_DACb(3)	Sets Pedestal DAC value driven to QIE – Channel b (msb)	Output	3.3V CMOS
35	DIEPDGND		N/C	Package leadframe die pad	gnd	0V
36	N/C		N/C			
37	N/C		N/C			
38	N/C		N/C			
39	N/C					
40	VDD	32	VDD		pwr/gnd	3.3V
41	VDDA	33	VDDA	Analog power	pwr/gnd	3.3V
42	GNDA	34	GNDA	Analog ground	pwr/gnd	0V
43	VSSD	35	VSSD		pwr/gnd	0V
44	VDDD	36	VDDD	Digital power	pwr/gnd	3.3V
45	VSSD	37	VSSD		pwr/gnd	0V
46	GNDD	38	GNDD	Ground	pwr/gnd	0V
47	RST_DLL*	39	RST_DLL*	DLL reset. Active low	Input	3.3V CMOS

48	RST_CCA*	40	RST_CCA*	Chip reset. Active low	Input	3.3V CMOS
49	BC_ZERO	41	BC_ZERO	BC ZERO Marker, comes once an orbit	Input	3.3V CMOS
50	RBX_A(2)	42	RBX_A(2)	RBXbus Address line	Input	3.3V CMOS
51	RBX_A(3)	43	RBX_A(3)	RBXbus Address line	Input	3.3V CMOS
52	RBX_A(4)	44	RBX_A(4)	RBXbus Address line	Input	3.3V CMOS
53	RBX_A(5)	45	RBX_A(5)	RBXbus Address line	Input	3.3V CMOS
54	RBX_A(6)	46	RBX_A(6)	RBXbus Address line	Input	3.3V CMOS
55	RBX_A(7)	47	RBX_A(7)	RBXbus Address line	Input	3.3V CMOS
56	RBX_CLK	48	RBX_CLK	RBXbus Serial Clock	Input	3.3V CMOS
57	VDDD	49	VDDD	Digital power	pwr/gnd	3.3V
58	VSSD	50	VSSD		pwr/gnd	0V
59	GNDD	51	GNDD		pwr/gnd	0V
60	RBX_DATA	52	RBX_DATA	RBXbus Data	Input/O utput	3.3V CMOS - Open Drain
61	GND	53	GND		pwr/gnd	0V
62	VSSD	54	VSSD		pwr/gnd	0V
63	VDD	55	VDD		pwr/gnd	3.3V
64	N/C		N/C			
65	N/C		N/C			
66	N/C		N/C			
67	N/C					
68	N/C					
69	+2.5V	56	+2.5V	Digital power	pwr/gnd	2.5V CMOS
70	DATA	57	DATA	Data word flag sent to serializer	Output	2.5V CMOS
71	CNTRL	58	CNTRL	Control word flag sent to serializer	Output	2.5V CMOS
72	SEND_FF	59	SEND_FF	Active high signal which enables sending of sync/idle words	Output	2.5V CMOS
73	TX_DATA(18)	60	TX_DATA(18)	Data sent to serializer	Output	2.5V CMOS
74	TX_DATA(17)	61	TX_DATA(17)	Data sent to serializer	Output	2.5V CMOS
75	TX_DATA(16)	62	TX_DATA(16)	Data sent to serializer	Output	2.5V CMOS
76	TX_DATA(15)	63	TX_DATA(15)	Data sent to serializer	Output	2.5V CMOS
77	TX_DATA(14)	64	TX_DATA(14)	Data sent to serializer	Output	2.5V CMOS
78	TX_DATA(13)	65	TX_DATA(13)	Data sent to serializer	Output	2.5V CMOS

79	TX_DATA(12)	66	TX_DATA(12)	Data sent to serializer	Output	2.5V CMOS
80	TX_DATA(11)	67	TX_DATA(11)	Data sent to serializer	Output	2.5V CMOS
81	TX_DATA(10)	68	TX_DATA(10)	Data sent to serializer	Output	2.5V CMOS
82	TX_DATA(9)	69	TX_DATA(9)	Data sent to serializer	Output	2.5V CMOS
83	TX_DATA(8)	70	TX_DATA(8)	Data sent to serializer	Output	2.5V CMOS
84	TX_DATA(7)	71	TX_DATA(7)	Data sent to serializer	Output	2.5V CMOS
85	TX_DATA(6)	72	TX_DATA(6)	Data sent to serializer	Output	2.5V CMOS
86	TX_DATA(5)	73	TX_DATA(5)	Data sent to serializer	Output	2.5V CMOS
87	TX_DATA(4)	74	TX_DATA(4)	Data sent to serializer	Output	2.5V CMOS
88	TX_DATA(3)	75	TX_DATA(3)	Data sent to serializer	Output	2.5V CMOS
89	TX_DATA(2)	76	TX_DATA(2)	Data sent to serializer	Output	2.5V CMOS
90	TX_DATA(1)	77	TX_DATA(1)	Data sent to serializer (lsb)	Output	2.5V CMOS
91	GND	78	GND		pwr/gnd	0V
92	VSSD	79	VSSD		pwr/gnd	0V
93	VDD	80	VDD		pwr/gnd	3.3V
94	<b>QIE_EXPA(1)*</b>	81	<b>QIE_EXPA(0)*</b>	Bit(1)* of QIE exponent – Channel a (msb)	Input	LVDS
95	<b>QIE_EXPA(1)</b>	82	<b>QIE_EXPA(0)</b>	Bit(1) of QIE exponent – Channel a (msb)	Input	LVDS
96	<b>QIE_EXPA(0)*</b>	83	<b>QIE_EXPA(1)*</b>	Bit(0)* of QIE exponent – Channel a (lsb)	Input	LVDS
97	<b>QIE_EXPA(0)</b>	84	<b>QIE_EXPA(1)</b>	Bit(0) of QIE exponent – Channel a (lsb)	Input	LVDS
98	<b>QIE_MANTa(4)*</b>	85	<b>QIE_MANTa(0)*</b>	Bit(4)* of QIE Mantissa – Channel a (msb)	Input	LVDS
99	<b>QIE_MANTa(4)</b>	86	<b>QIE_MANTa(0)</b>	Bit(4) of QIE Mantissa – Channel a (msb)	Input	LVDS
100	N/C		N/C			
101	N/C		N/C			
102	N/C					
103	N/C					
104	<b>QIE_MANTa(3)*</b>	87	<b>QIE_MANTa(1)*</b>	Bit(3)* of QIE Mantissa – Channel a	Input	LVDS
105	<b>QIE_MANTa(3)</b>	88	<b>QIE_MANTa(1)</b>	Bit(3) of QIE Mantissa – Channel a	Input	LVDS
106	<b>QIE_MANTa(2)*</b>	89	<b>QIE_MANTa(2)*</b>	Bit(2)* of QIE Mantissa – Channel a	Input	LVDS
107	<b>QIE_MANTa(2)</b>	90	<b>QIE_MANTa(2)</b>	Bit(2) of QIE Mantissa – Channel a	Input	LVDS
108	<b>QIE_MANTa(1)*</b>	91	<b>QIE_MANTa(3)*</b>	Bit(1)* of QIE Mantissa – Channel a	Input	LVDS
109	<b>QIE_MANTa(1)</b>	92	<b>QIE_MANTa(3)</b>	Bit(1) of QIE Mantissa –	Input	LVDS

				Channel a		
110	<b>QIE_MANTa(0)*</b>	93	<b>QIE_MANTa(4)*</b>	Bit(0)* of QIE Mantissa – Channel a (lsb)	Input	LVDS
111	<b>QIE_MANTa(0)</b>	94	<b>QIE_MANTa(4)</b>	Bit(0) of QIE Mantissa – Channel a (lsb)	Input	LVDS
112	<b>QIE_CAPIDa(0)*</b>	95	<b>QIE_CAPIDa(0)*</b>	Bit(0)* of QIE Capacitor ID – Channel a (lsb)	Input	LVDS
113	<b>QIE_CAPIDa(0)</b>	96	<b>QIE_CAPIDa(0)</b>	Bit(0) of QIE Capacitor ID – Channel a (lsb)	Input	LVDS
114	<b>QIE_CAPIDa(1)*</b>	97	<b>QIE_CAPIDa(1)*</b>	Bit(1)* of QIE Capacitor ID – Channel a (msb)	Input	LVDS
115	<b>QIE_CAPIDa(1)</b>	98	<b>QIE_CAPIDa(1)</b>	Bit(1) of QIE Capacitor ID – Channel a (msb)	Input	LVDS
116	<b>QIE_EXPb(1)*</b>	99	<b>QIE_EXPb(0)*</b>	Bit(1)* of QIE exponent – Channel b (msb)	Input	LVDS
117	<b>QIE_EXPb(1)</b>	100	<b>QIE_EXPb(0)</b>	Bit(1) of QIE exponent – Channel b (msb)	Input	LVDS
118	<b>QIE_EXPb(0)*</b>	101	<b>QIE_EXPb(1)*</b>	Bit(0)* of QIE exponent – Channel b (lsb)	Input	LVDS
119	<b>QIE_EXPb(0)</b>	102	<b>QIE_EXPb(1)</b>	Bit(0) of QIE exponent – Channel b (lsb)	Input	LVDS
120	<b>QIE_MANTb(4)*</b>	103	<b>QIE_MANTb(0)*</b>	Bit(4)* of QIE Mantissa – Channel b (msb)	Input	LVDS
121	<b>QIE_MANTb(4)</b>	104	<b>QIE_MANTb(0)</b>	Bit(4) of QIE Mantissa – Channel b (msb)	Input	LVDS
122	<b>QIE_MANTb(3)*</b>	105	<b>QIE_MANTb(1)*</b>	Bit(3)* of QIE Mantissa – Channel b	Input	LVDS
123	<b>QIE_MANTb(3)</b>	106	<b>QIE_MANTb(1)</b>	Bit(3) of QIE Mantissa – Channel b	Input	LVDS
124	<b>QIE_MANTb(2)*</b>	107	<b>QIE_MANTb(2)*</b>	Bit(2)* of QIE Mantissa – Channel b	Input	LVDS
125	<b>QIE_MANTb(2)</b>	108	<b>QIE_MANTb(2)</b>	Bit(2) of QIE Mantissa – Channel b	Input	LVDS
126	<b>QIE_MANTb(1)*</b>	109	<b>QIE_MANTb(3)*</b>	Bit(1)* of QIE Mantissa – Channel b	Input	LVDS
127	<b>QIE_MANTb(1)</b>	110	<b>QIE_MANTb(3)</b>	Bit(1) of QIE Mantissa – Channel b	Input	LVDS
128	N/C					

\*"LVDS like" signals will be generated by the QIE. These signals will not drive a cable, but will be sufficient to drive the couple of inches of traces we expect to use. The signal levels are compatible with LVDS levels and may be received and re-driven with a commercial LVDS receiver is desired.

# **APPENDIX A**

## **Simulation ideas**

The following are some suggested examples of behavioral simulations which should be done on the CCA ASIC. They are meant to provide a starting point for functional test, and do not provide a full simulation.

**Check RBXbus Interface – using RBXbus protocol,**

**RBXbus Block Read/Write**

Write 0x00 to the RBX\_Pointer register

Write a block of 28 words to the RBX\_Data register

Write 0x00 to the RBX\_Pointer register

Read a block of 28 words from the RBX\_Data register

Verify data read is the same as data written

Also test at beginning address other than 0x00

**RBXbus Single Word Read/Write**

Write an arbitrary address (0x00 – 0x1B) to the RBX\_Pointer register

Write a single data word to the RBX\_Data register

Read a single data word from the RBX\_Data register

Verify data read is the same as data written

### Check the function of the Test Pulses

Assume free running Master Clock at 40.08 MHz  
 Assume Beam\_Zero coming once every 3564 Master clocks

Write 0x04 to the RBX_Pointer register	“points to DLL Tap Select 0
Write the following block of data (in Block mode) to RBX_Data register:	
0x04 <sup>1</sup>	“phases QIE_CLKa
0x0A <sup>1</sup>	“phases QIE_CLKb
0x23 <sup>1</sup>	“set Bunch counter match (LSB)
0x01 <sup>1</sup>	“set Bunch counter match (MSB)
Write 0x00 to the RBX_Pointer register	“points to Control Register
Write 0x01 to the RBX_Data register	“sets Enable Test Pulse

**Expected results:** Should see test pulses issued when the Bunch counter matches 0x0123<sup>1</sup>. The test pulses should be one clock cycle long, with TEST\_PULSEa sync'd to QIE\_CLKa, and TEST\_PULSEb sync'd to QIE\_CLKb. The TEST\_PULSE signals should be repeated once every orbit until you write

Write 0x00 to the RBX_Pointer register	“points to Control Register
Write 0x00 to the RBX_Data register	“clears Enable Test Pulse

### Also test Polarity Control of the Test Pulses.

<sup>1</sup> This is an arbitrary value. Simulation can vary this.

### Signals which should be observed:

- Master Clock
- QIE\_CLKa
- QIE\_CLKb
- TEST\_PULSE\_TRIGa
- TEST\_PULSE\_TRIGb
- QIE\_RESETa
- QIE\_RESEtb
- CAP\_IDa(1:0)
- CAP\_IDb(1:0)

\* would like to see QIE\_CLKa,b phased into all groupings.

### Check the ability to send Test Patterns

Assume free running Master Clock at 40.08 MHz

Assume Beam\_Zero coming once every 3564 Master clocks

Write 0x08 to the RBX_Pointer register	“points to Test Pattern byte register
Write the following block of data (in Block mode) to RBX_Data register:	
0x01 <sup>1</sup>	“write Test Pattern byte
0x02 <sup>1</sup>	“write Test Pattern byte
0x03 <sup>1</sup>	“write Test Pattern byte
0x04 <sup>1</sup>	“write Test Pattern byte
0x05 <sup>1</sup>	“write Test Pattern byte
0x06 <sup>1</sup>	“write Test Pattern byte
0x07 <sup>1</sup>	“write Test Pattern byte
0x08 <sup>1</sup>	“write Test Pattern byte
0x09 <sup>1</sup>	“write Test Pattern byte
0x0A <sup>1</sup>	“write Test Pattern byte
0x0B <sup>1</sup>	“write Test Pattern byte
0x0C <sup>1</sup>	“write Test Pattern byte
0x0D <sup>1</sup>	“write Test Pattern byte
0x0E <sup>1</sup>	“write Test Pattern byte
0x0F <sup>1</sup>	“write Test Pattern byte
0x10 <sup>1</sup>	“write Test Pattern byte
0x11 <sup>1</sup>	“write Test Pattern byte
0x12 <sup>1</sup>	“write Test Pattern byte
0x13 <sup>1</sup>	“write Test Pattern byte
0x14 <sup>1</sup>	“write Test Pattern byte
Write 0x00 to the RBX_Pointer register	“points to Control Register
Write 0x02 to the RBX_Data register	“sets Orbit Data source

**Expected results:** Should see the 20 byte test pattern sent to the serializer after the receipt of Beam\_Zero. The test pattern should be packed into the Orbit Message in a format described in the **Orbit Message** description as well as Fill Frames.

Additionally, a QIE\_RESET signal should be produced with the receipt of Beam\_zero.

The Test Pattern should be repeated once every orbit until you write

Write 0x00 to the RBX_Pointer register	“points to Control Register
Write 0x00 to the RBX_Data register	“clears Enable Test Pulse

**Check the ability set Calibration Mode**

Assume free running Master Clock at 40.08 MHz  
 Assume Beam\_Zero coming once every 3564 Master clocks

Write 0x00 to the RBX\_Pointer register      “points to Control register  
 Write 0x20 to the RBX\_Data register      “sets Enable Calibration Mode bit

**Expected results:** Should see the CALIB\_MODE output bit go high

**Check the ability set Fill Frame command to Serializer**

Assume free running Master Clock at 40.08 MHz  
 Assume Beam\_Zero coming once every 3564 Master clocks

Write 0x00 to the RBX\_Pointer register      “points to Control register  
 Write 0x40 to the RBX\_Data register      “sets Send Fill Frame bit

**Expected results:** Should see the Send\_Fill\_Frames output bit go high

**Check the ability Force the QIE Range**

Assume free running Master Clock at 40.08 MHz  
 Assume Beam\_Zero coming once every 3564 Master clocks

Write 0x00 to the RBX\_Pointer register      “points to Control register  
 Write 0x38<sup>1</sup> to the RBX\_Data register      “sets Range bits

**Expected results:** Should see the Enable\_QIE\_Range output bit go high. Should also see highs on the QIE\_RANGE(1:0) bits.

<sup>1</sup> This is an arbitrary value. Simulation can vary this.

**Check the ability set Pedestal DAC bits**

## Check Normal Running Mode

Assume free running Master Clock at 40.08 MHz  
 Assume Beam\_Zero coming once every 3564 Master clocks  
 Assume data coming from 2 QIE channels

Write 0x00 to the RBX_Pointer register	“points Control Register
Write the following block of data (in Block mode) to RBX_Data register:	
0x00	“sets Control register to default
0x00	“sets Alignment control 1
0x00	“sets Alignment control 2
0x00 <sup>1</sup>	“sets Ped DAC
0x04 <sup>1</sup>	“sets QIE_CLKa phase
0x0A <sup>1</sup>	“sets QIE_CLKb phase
0x23 <sup>1</sup>	“sets TP BC Match reg (LSB)
0x01 <sup>1</sup>	“sets TP BC Mat reg (MSB)

**Expected results:** Should see Orbit messages (format described in **Orbit Message** section) and QIE\_Reset issued when Beam\_zero occurs. Should see QIE data being packed and sent during non “Orbit message” cycles.

Should force errors on Cap Ids and Bunch crossing to test Orbit message error flags.

Error Conditions:

Issue a B0 followed by another B0 50 clock ticks later.

If sending Orbit Message with test pattern, and writing test pattern bytes over RBXbus simultaneously, make sure CCA does not become lost in state machine.

What happens, if CCA never gets B0? We would like to have it in normal run mode of sending QIE data.

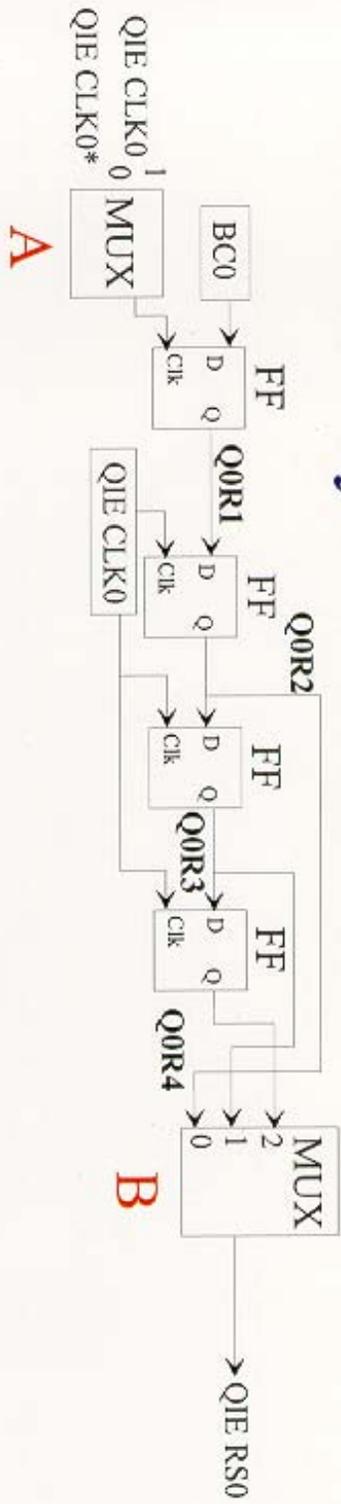
<sup>1</sup> This is an arbitrary value. Simulation can vary this.

# **APPENDIX B**

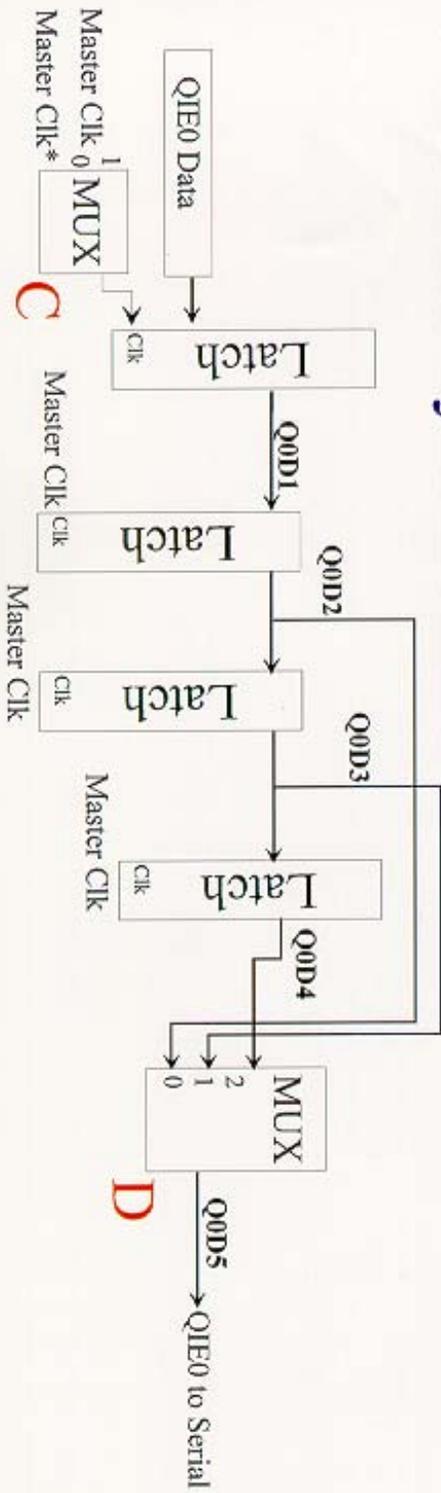
## **Channel Alignment**

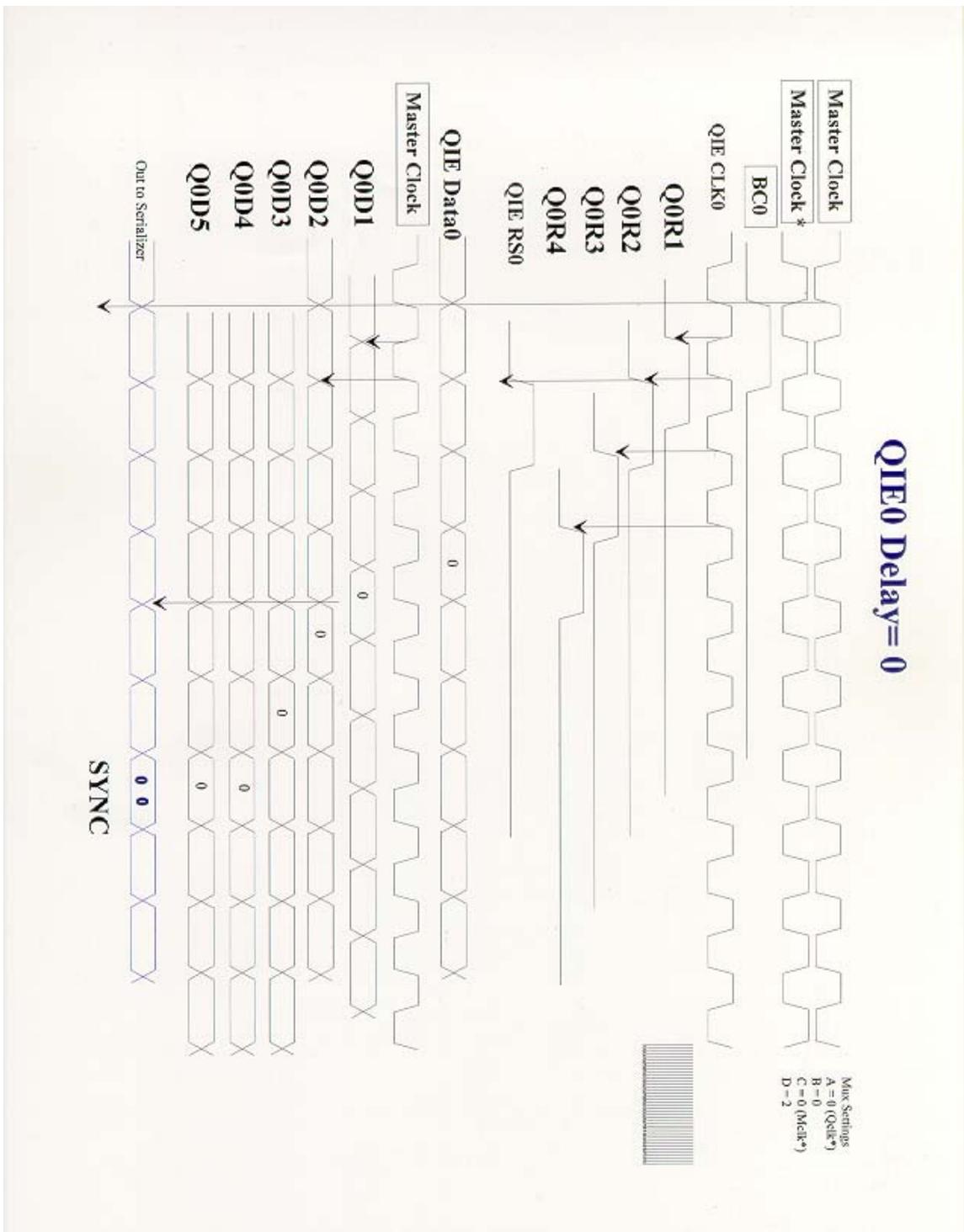
**The following design and time settings are meant as a solution guide towards the problem of timing in QIE channels and resynchronizing the data to the LHC\_Clock. The solution presented should allow QIE channels which are as much as 58ns out of sync, to be aligned properly and re-phased to the LHC\_Clock.**

# CCA Reset Synchronizer

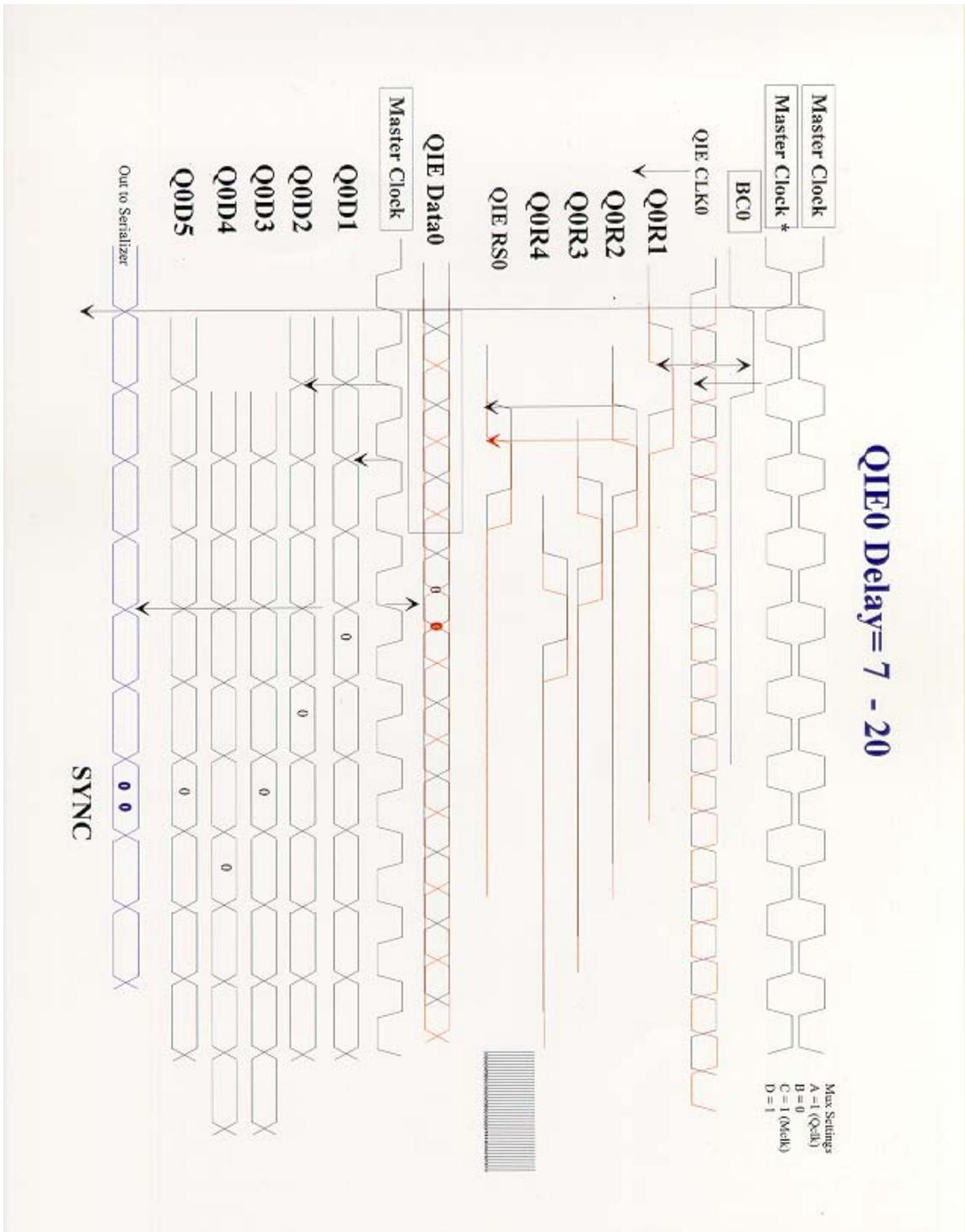


# CCA Data Synchronizer

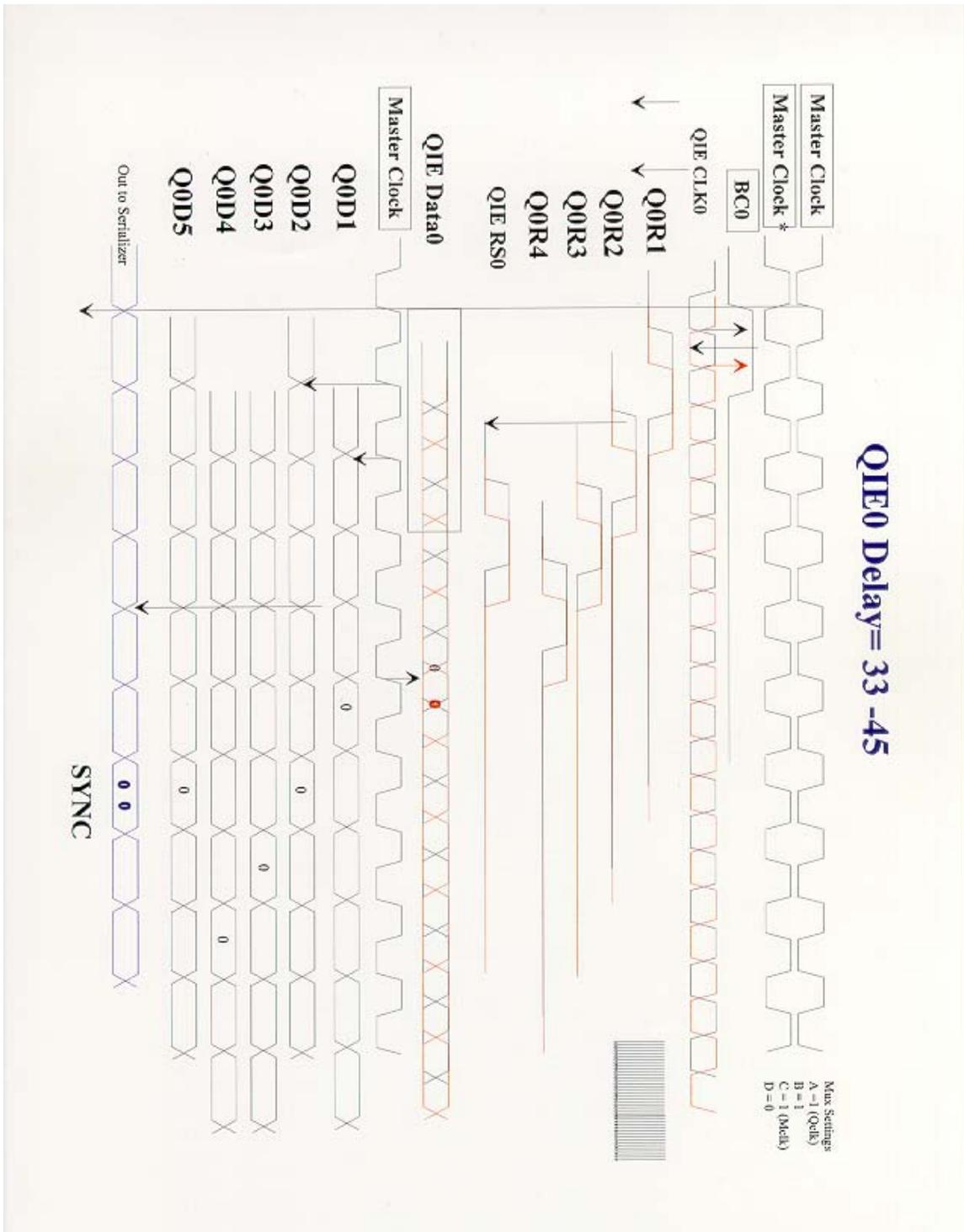


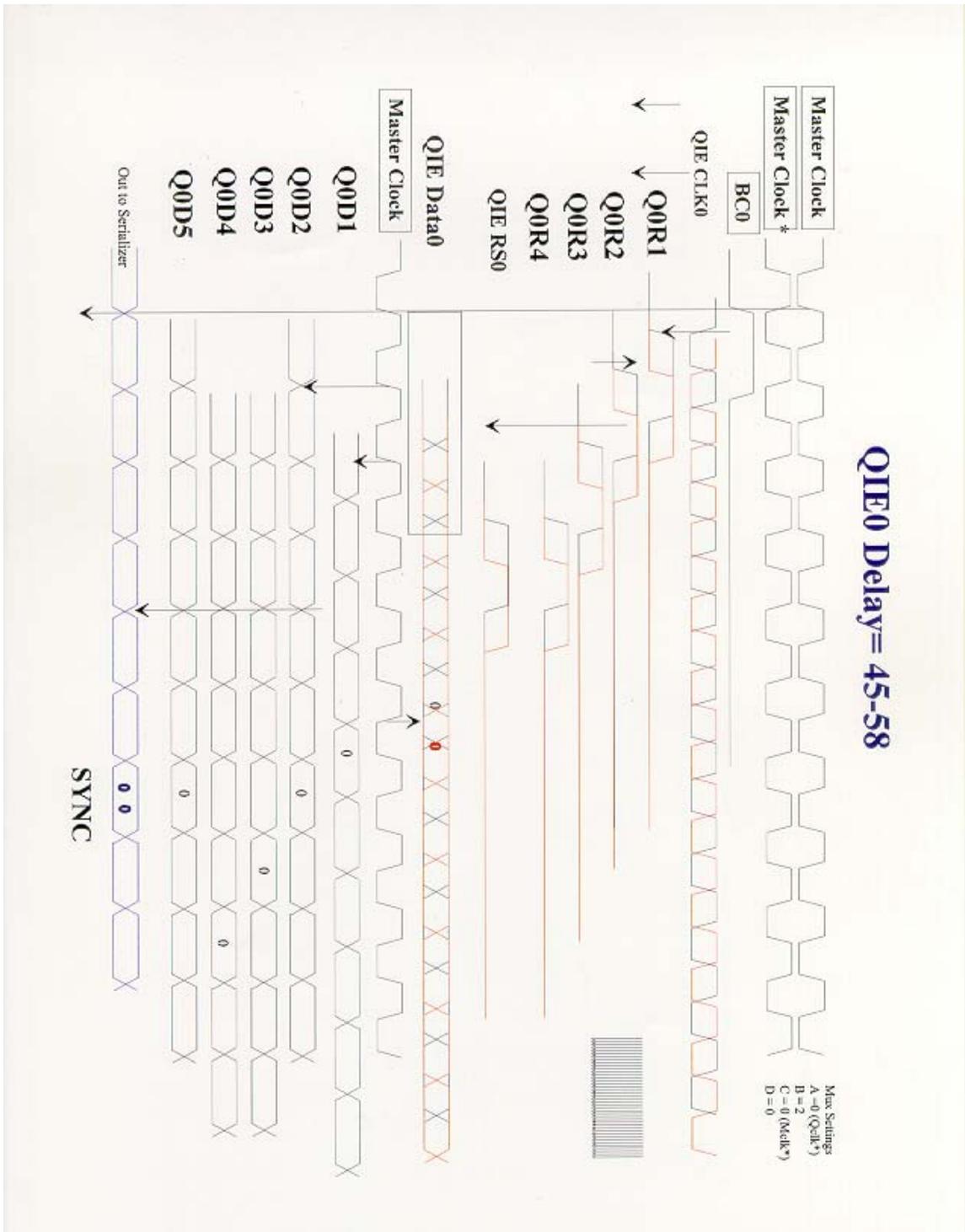






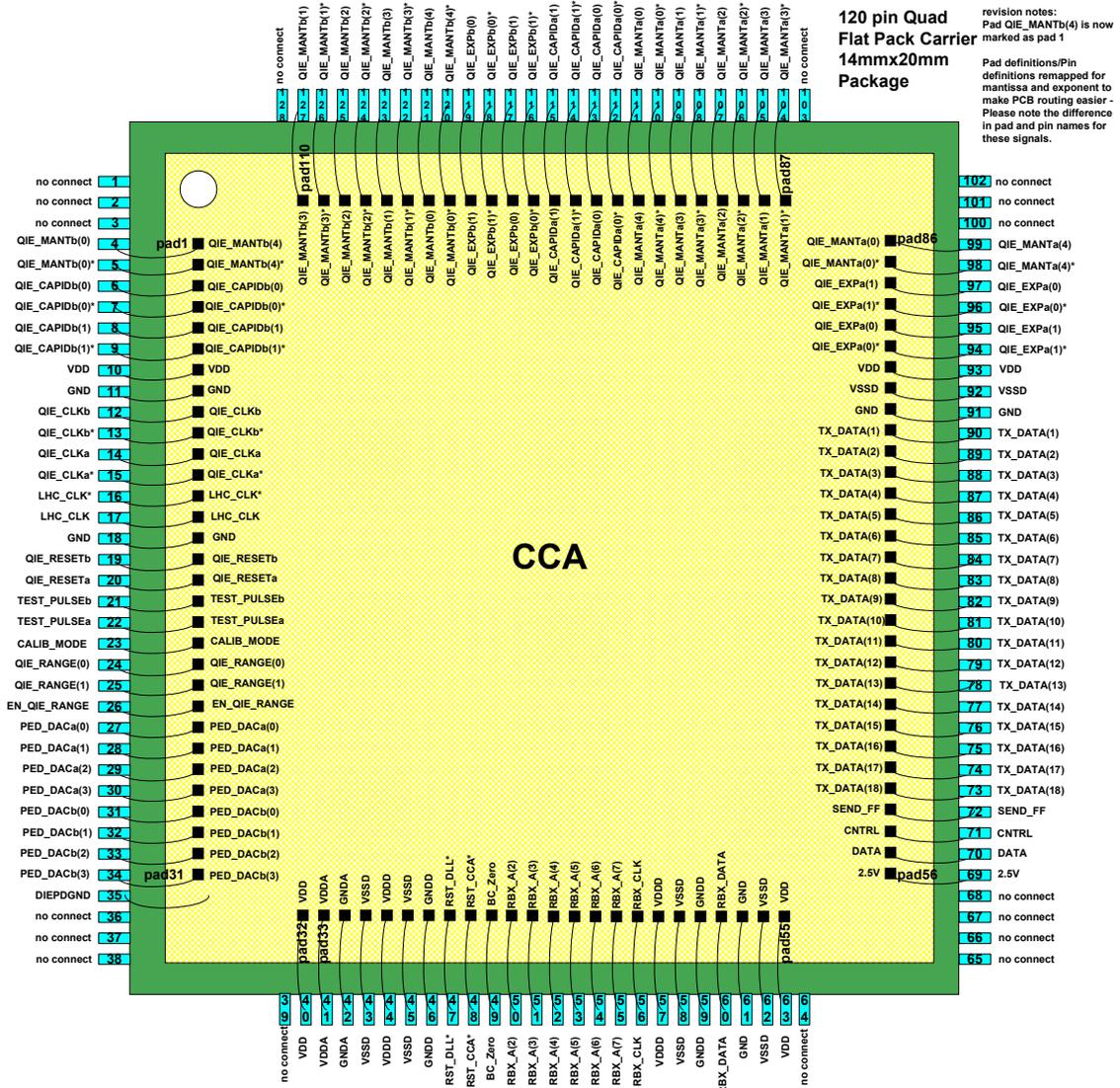






## **APPENDIX C**

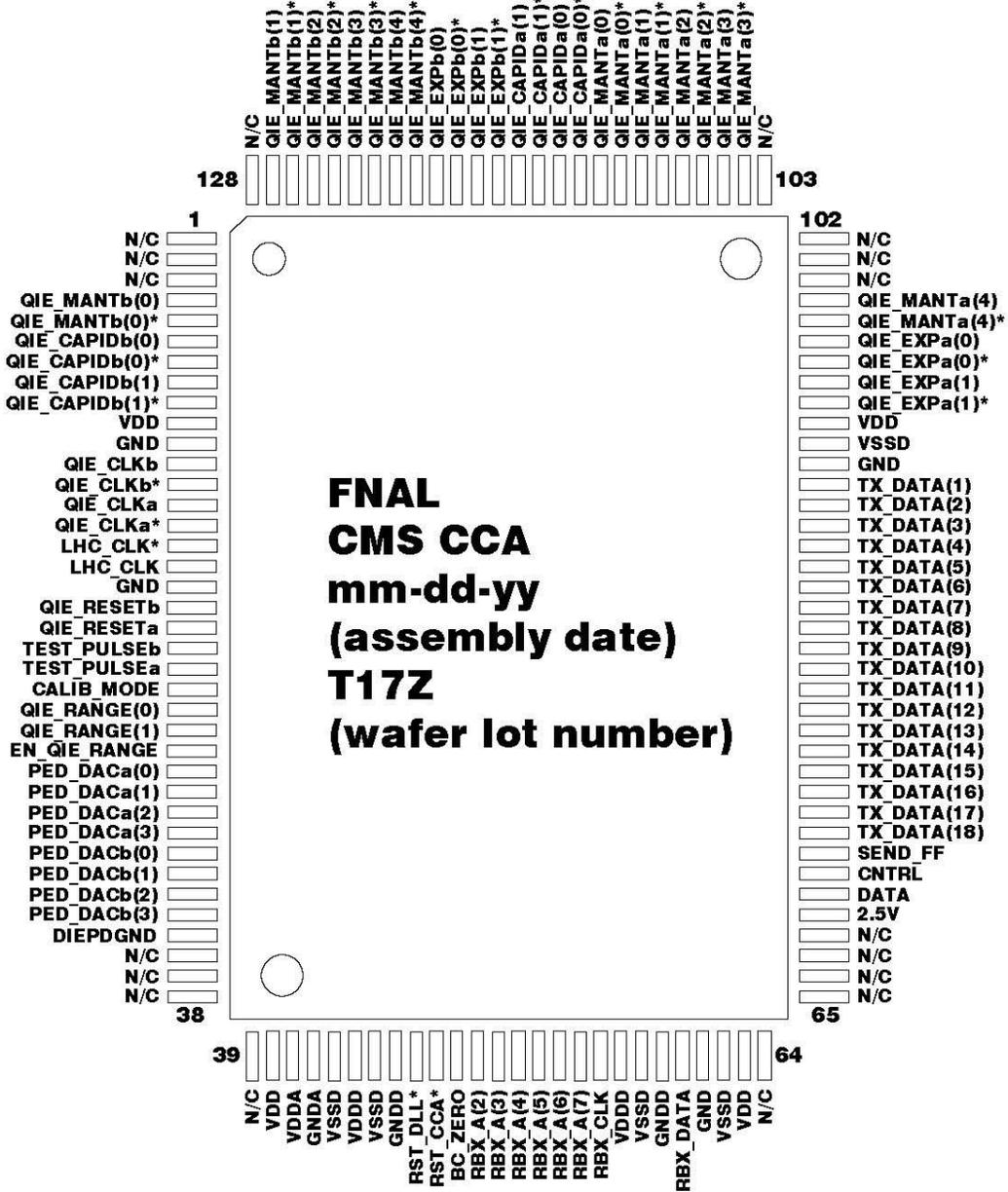
### **Pin/Pad Assignments Production Chip**



# CMS CCA

03/07/02

## 128-Lead QFP Package Pin Assignments



# CMS CCA

03/07/02

## 128-Lead QFP Package Bonding Diagram

